Quick and Dirty Search Engine with PHP and MySQL

By Clay Johnson

Table of Contents

Introduction.....1

Searching the Table.....3

Introduction

So you've got a dynamic site, filled with all sorts of user inputs, whether it be a 'phorum', or like my own site at http://www.knowpost.com. ht://dig will take care of indexing and searching your html pages, but if you are like me, you have very few html pages, and most of your "content" resides in BLOBs in your database. You can't do anything useful using a like %searchword% query, it just isn't coming back relevant.

There has to be a better way, and indeed there is, with a few easy steps. Here's how to slap one together:

Noise Reduction
The first problem with your content is that it is filled with clunky "noisewords," like "a,the,where,look" Things that are there to help us humans to communicate, but really don't have anything to do with relevance. We gotta get rid of those. I've included a big list of noisewords (noisewords.txt) for you to use, modify or mutilate. Essentially, what we're trying to do here is get all those noisewords out of your data, and build a table with two columns, the word, and its indicator (the content associated with it). We want something that will eventually look like this:

```
+------+------------+
| qid  | word       |
+------+------------+
|    6 | links      |
|    5 | Fire       |
|    5 | topics     |
|    5 | related    |
|    5 | Shakespeare|
|    4 | people     |
|    4 | Knowpost   |
|    3 | cuba       |
|    3 | cigar      |
+------+------------+
```

Lets create our table now:

```
mysql> CREATE TABLE search_table( word VARCHAR(50), qid INT )
```

Next, since you want to make all your data compatible, not just new data, we need to grab your sticky blobs, and their identifiers out of your database:

```
<? $query = "SELECT blob,identifier FROM your_table"; $result = mysql_query($query);
$number = mysql_numrows($result); $j = 0; WHILE ($j < $number){ /* Your "blob" */
$body = mysql_result($result,$j,"blob"); /*Your "identifier" */ $qid =
mysql_result($result,$j,"qid"); /* Open the noise words into an array */ $noise_words =
file("noisewords.txt"); $filtered = $body; /* Got to put a space before the first word in the
body, so that we can recognize the word later */ $filtered = ereg_replace("^"," ",$filtered); /*
Now we suck out all the noisewords, and transform whats left into an array */ /* Brought to
you by poor ereg coding! */ for ($i=0; $i<count($noise_words); $i++) { $filterword =
trim($noise_words[$i]); $filtered = eregi_replace(" $filterword "," ",$filtered); } $filtered =
trim($filtered); $filtered = addslashes($filtered); $querywords = ereg_replace(",","",$filtered);
```

```
$querywords = ereg_replace(" ",",",$querywords); $querywords =
ereg_replace("?","",$querywords); $querywords = ereg_replace("(","",$querywords);
$querywords = ereg_replace(")","",$querywords); $querywords =
ereg_replace(".","",$querywords); $querywords = ereg_replace(",","'",'",$querywords);
$querywords = ereg_replace("^","'",$querywords); $querywords =
ereg_replace("$","'",$querywords); /* We should now have something that looks like
'Word1','Word2','Word3' so lets turn it into an array */ $eachword = explode(",",
$querywords); /* and finally lets go through the array, and place each word into the database,
along with its identifier */ for ($k=0; $k<count($eachword); $k++){ $inputword = "INSERT
INTO search_table VALUES($eachword[$k],$qid)"; mysql_query($inputword); } /* Get the
next set of data */ $j++; } ?>
```

That script just handles your old data. You'll want to include a similar function to strip the noisewords out for every time new information comes into your database, through user input, your input, etc... so that your search engine is updated on the fly.

Searching the Table

Now you have an easy to−use table of keywords and their associations. How do you query this table? Here's what I do:

First I format each searchterms passed into the script as 'word1','word2','word3' and stick it in a string called $querywords.

Then I throw them into this SQL query:

```
SELECT count(search_table.word) as score, search_table.qid,your_table.blob FROM
search_table,your_table WHERE your_table.qid = search_table.qid AND search_table.word
IN($querywords) GROUP BY search_table.qid ORDER BY score DESC";
```

Set that query to $search, and print out the results like so:

```
<? $getresults = mysql_query($search); $resultsnumber = mysql_numrows($getresults); IF
($resultsnumber == 0){ PRINT "Your search returned no results. Try other keyword(s)."; }
ELSEIF ($resultsnumber > 0){ PRINT "Your search returned $resultsnumber
results<BR>Listing them in order of relevance<BR><BR>"; for($count = 0;
$count<$resultsnumber; $count++){ $body = mysql_result($getresults,$count,"blob"); $qid =
mysql_result($getresults,$count,"qid"); //tighten up the results $body2print = substr($body, 0,
100); $cnote = $count+1; PRINT "$cnote. <a href=yourcontent.php3?qid=$qid>
<i>$body2print...</i></a><BR>"; } } ?>
```

Presto, you've got keyword searching for your database, complete with relevancy ranking. It may not be Google or altavista.

It may not support all those fancy boolean operators, or excite's (*cough*) conceptual mapping technology. But it works, its quick and enough to handle your user's demand.